Deep-Learning Traffic Flow Prediction for Forecasting Performance Measurement of Public Transportation Systems

January 2020

A White Paper from the Pacific Southwest Region University Transportation Center

Luan Tran, Ph.D. Student, University of Southern California

Min Mun, Research Staff, University of Southern California

Yao Yi Chiang, Co-Principal Investigator, University of Southern California

Cyrus Shahabi, Principal Investigator, University of Southern California

TECHNICAL REPORT DOCUMENTATION PAGE

<table>
<tr><td>1. Report No.
PSR-18-10</td><td colspan="3">2. Government Accession No.
N/A</td><td colspan="3">3. Recipient's Catalog No.
N/A</td></tr>
<tr><td colspan="4" rowspan="2">4. Title and Subtitle
Deep-Learning Traffic Flow Prediction for Forecasting Performance Measurement of Public Transportation Systems</td><td colspan="3">5. Report Date
January 31, 2020</td></tr>
<tr><td colspan="3">6. Performing Organization Code
N/A</td></tr>
<tr><td colspan="4">7. Author(s)
Cyrus Shahabi, Principal Investigator, USC https://orcid.org/0000-0001-9118-0681
Luan Tran, Ph.D. Student, USC
Min Mun, Research Staff, USC
Yao Yi Chiang, Co-Principal Investigator, USC</td><td colspan="3">8. Performing Organization Report No.
PSR-18-10 TO-001</td></tr>
<tr><td colspan="4" rowspan="2">9. Performing Organization Name and Address
University of Southern California
650 Childs Way, RGL 216
Los Angeles, CA 90089</td><td colspan="3">10. Work Unit No.
N/A</td></tr>
<tr><td colspan="3">11. Contract or Grant No.
USDOT Grant 69A3551747109
Caltrans Grant 65A0674, Task Order 001</td></tr>
<tr><td colspan="4" rowspan="2">12. Sponsoring Agency Name and Address
Caltrans DRISI
1727 30th Street, MS 82
Sacramento, CA 95816</td><td colspan="3">13. Type of Report and Period Covered
Final report (Feb. 2019–Jan. 2020)</td></tr>
<tr><td colspan="3">14. Sponsoring Agency Code
USDOT OST-R</td></tr>
<tr><td colspan="7">15. Supplementary Notes
Report URL: https://www.metrans.org/research/deep-learning-traffic-flow-prediction-for-forecasting-performance-measurement-of-public-transportation-systems</td></tr>
<tr><td colspan="7">16. Abstract
In this project, we developed a deep learning approach for traffic flow forecasting and bus arrival time estimation in Los Angeles. First, we developed a novel Graph Convolutional Recurrent Neural Network (GCRNN) to model and forecast traffic flows at different spatial and temporal resolutions. Our GCRNN model considers not only the location of traffic sensors but also their relationships (i.e., topological dependency) in space, which was critical to achieving the best performance for all forecasting horizons compared to the existing methods. Next, we implemented a Geo-Convolution Long Short-Term Memory (Geo-Conv LSTM) framework to model bus Estimated Time of Arrival (ETA) by incorporating the traffic flow predictions of our GCRNN. Using the real-world traffic sensor datasets archived in our data warehouse, we showed that our proposed bus ETA model is more accurate than the existing method, Gradient Boosted Decision Tree (GBDT), by 27% in estimating bus travel time. Lastly, we deployed both models as web applications so that users can access traffic prediction data and check bus arrival times to a destination location from a starting point.</td></tr>
<tr><td colspan="3">17. Key Words
Forecasting; Traffic analysis zones; Traffic flow; Traffic flow rate; Public Transportation Information Sharing and Analysis Center; Passenger Transportation; Public Transportation; Transportation</td><td colspan="4">18. Distribution Statement
No restrictions.</td></tr>
<tr><td colspan="2">19. Security Classif. (of this report)
Unclassified</td><td colspan="3">20. Security Classif. (of this page)
Unclassified</td><td>21. No. of Pages
36</td><td>22. Price
N/A</td></tr>
</table>

Form DOT F 1700.7 (8-72) Reproduction of completed page authorized

Pacific Southwest Region UTC
University Transportation Center

TABLE OF CONTENTS

About the Pacific Southwest Region University Transportation Center

The Pacific Southwest Region University Transportation Center (UTC) is the Region 9 University Transportation Center funded under the US Department of Transportation's University Transportation Centers Program. Established in 2016, the Pacific Southwest Region UTC (PSR) is led by the University of Southern California and includes seven partners: Long Beach State University; University of California, Davis; University of California, Irvine; University of California, Los Angeles; University of Hawaii; Northern Arizona University; Pima Community College.

The Pacific Southwest Region UTC conducts an integrated, multidisciplinary program of research, education and technology transfer aimed at improving the mobility of people and goods throughout the region. Our program is organized around four themes: 1) technology to address transportation problems and improve mobility, 2) improving mobility for vulnerable populations, 3) Improving resilience and protecting the environment, and 4) managing mobility in high growth areas.

U.S. Department of Transportation (USDOT) Disclaimer

California Department of Transportation (CALTRANS) Disclaimer

Disclosure

Shahabi (PI), Chiang (co-PI), Mun (Research Staff) and Tran (PhD student) conducted this research titled, "Deep-Learning Traffic Flow Prediction for Forecasting Performance Measurement of Public Transportation Systems" at the Computer Science Department, Viterbi School of Engineering, University of Southern California. The research took place from February 1st, 2019 to January 31st, 2020 and was funded from the California Department of Transportation - Caltrans in the amount of $100,000.00. The research was conducted as part of the Pacific Southwest Region University Transportation Center research program.

Acknowledgement

We are grateful to Chaoyang He, Haoze Yang, Matthew Lim, Nathan Huh, and Jonah Yamato for their help on deploying our deep-learning-based Bus ETA model.

Abstract

Los Angeles is ranked the most congested city in the U.S. with a typical half-hour commute taking 81% longer during evening peak periods and 60% longer during the morning peak. These traffic congestions result in a large social and economic detriment and raise serious concern for drivers and transportation agencies. Therefore, increasing ridership of public transportations and hence reducing traffic congestions has been one of the primary objectives for transportation agencies and policymakers. Previously, many researchers have worked on estimating historical performance measurements of public transportation systems. Beyond historical performance measurements, accurate predictive analysis of performance reliability helps to manage rider expectations as well as to provide a powerful tool for transportation agencies to coordinate the public transportation vehicles. For the first time, there is a unique opportunity to use data-driven approaches that analyze big datasets collected from transportation systems to understand the factors causing traffic congestions and in turn, help to forecast the performance reliability of public transportation vehicles.

In this project, we developed a deep learning approach for traffic flow forecasting and bus arrival time estimation in Los Angeles. First, we developed a novel Graph Convolutional Recurrent Neural Network (GCRNN) to model and forecast traffic flows at different spatial and temporal resolutions. Our GCRNN model considers not only the location of traffic sensors but also their relationships (i.e., topological dependency) in space, which was critical to achieving the best performance for all forecasting horizons compared to the existing methods. Next, we implemented a Geo-Convolution Long Short-Term Memory (Geo-Conv LSTM) framework to model bus Estimated Time of Arrival (ETA) by incorporating the traffic flow predictions of our GCRNN. Using the real-world traffic sensor datasets archived in our data warehouse, we showed that our proposed bus ETA model is more accurate than the existing method, Gradient Boosted Decision Tree (GBDT), by 27% in estimating bus travel time. Lastly, we deployed both models as web applications so that users can access traffic prediction data and check bus arrival times to a destination location from a starting point.

A Deep-Learning Technique for Bus Travel Time Prediction using Traffic Forecasting Measurements

Executive Summary

In 2015, 439 metropolitan areas experienced 7 billion vehicle-hours of delay, which is equivalent to 3 billion gallons in wasted fuel and $160 billion in lost productivity. These traffic congestions result in a large social and economic detriment to the U.S. and raise serious concern for drivers and transportation agencies. Therefore, increasing ridership of public transportations and hence reducing traffic congestions has been one of the primary objectives for transportation agencies and policymakers. Los Angeles usually tops the list of gridlock-plagued cities with an average of 80 hours of delay per commuter a year and improving ridership of public transportation like the public bus system is one of the primary objectives for the California Department of Transportation. Historical performance measurements of public transportation systems can help identify problems for improving ridership. Beyond historical performance measurements, accurate predictive analysis of performance reliability helps to manage rider expectations (e.g., will the bus be on time in the next 30 minutes?) as well as to provide a powerful tool for transportation agencies to coordinate the public transportation vehicles.

At USC's Integrated Media Systems Center (IMSC) with our partnership with Los Angeles Metropolitan Transportation Authority (LA Metro) and METRANS, we developed a big transportation data warehouse-Archived Traffic Data Management System (ADMS). ADMS fuses and analyzes a very large-scale and high-resolution (both spatial and temporal) traffic sensor data from different transportation authorities in Southern California, including California Department of Transportation (Caltrans), Los Angeles Department of Transportation (LADOT), California Highway Patrol (CHP), Long Beach Transit (LBT). This data set includes both inventory and real-time data with update rate as high as every 30 seconds for freeway and arterial traffic sensors (14,500 loop-detectors) covering 4,300 miles, 2,000 bus, and train automatic vehicle location (AVL), incidents such as accidents, traffic hazards and road closures reported (approximately 400 per day) by LAPD and CHP, and ramp meters. We have been continuously collecting and archiving datasets for the past 5 years. ADMS, with 11TB annual growth, is the largest traffic sensor data warehouse built so far in Southern California. Using this big traffic dataset, we have a unique opportunity to use data-driven approaches to understand the factors causing traffic congestions and in turn, help to forecast the performance reliability of public transportation vehicles.

In this project, we developed a reliability analysis system using Deep Learning (DL) techniques to forecast the future performances of the public bus system in Los Angeles. First, we developed a novel Graph Convolutional Recurrent Neural Network (GCRNN) to model and forecast traffic flows at different spatial (e.g., individual regions, road segments, or sensors) and the temporal (e.g., next 5 minutes and 30 minutes) resolutions. Our GCRNN model considers not only the location of traffic sensors but also their relationships (i.e., topological dependency) in space, which was critical to achieving the best performance for all forecasting horizons compared to the existing methods. Next, we implemented a Geo-Convolution Long Short-Term Memory (Geo-Conv LSTM) framework to model bus Estimated Time of Arrival (ETA) by incorporating the traffic flow predictions of our GCRNN. Using the real-world traffic sensor datasets archived in our data warehouse, we showed that our proposed bus ETA model is more accurate than the existing method, Gradient Boosted Decision Tree (GBDT), by 27% in estimating bus travel times. Lastly, we deployed both models as web applications so that users can access traffic prediction data and check bus arrival times to a destination location from a starting point.

1. Introduction

In 2015, 439 metropolitan areas experienced 7 billion vehicle-hours of delay, which is equivalent to 3 billion gallons in wasted fuel and $160 billion in lost productivity. According to the TomTom Traffic Index, Los Angeles is ranked the most congested city in the U.S., the 12th most congested city worldwide, with a typical half-hour commute taking 81% longer during evening peak periods and 60% longer during the morning peak. These traffic congestions result in a large social and economic detriment and raise serious concern for drivers and transportation agencies. Therefore, increasing ridership of public transportations and hence reducing traffic congestions has been one of the primary objectives for transportation agencies and policymakers. For example, improving the performance and reliability of public transportation vehicles has been one of the primary objectives for California Department of Transportation (Caltrans).

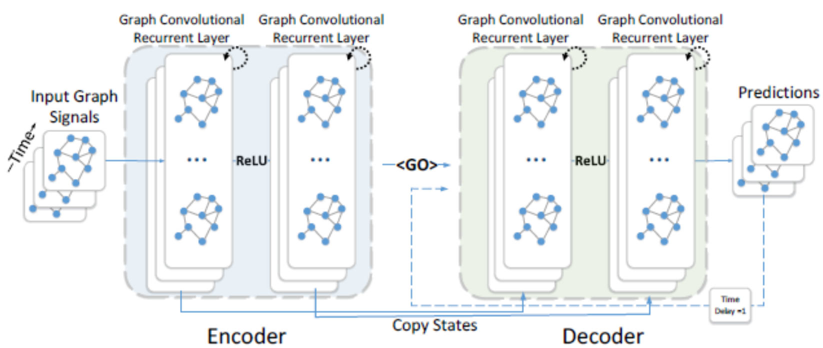
Historical performance measurements of public transportation systems can help identify problems and potential solutions for improving ridership. For example, historical trends of bus travel-time reliability and on-time performance can help city transportation agencies to quickly identify potential problems with existing bus routes, such as quantifying the delays in bus lines caused by constructions in the city or making informed policy decisions including rearranging bus timetables. Beyond historical performance measurements, accurate predictive analysis of performance reliability helps to manage rider expectations (e.g., will the bus be on time in the next 30 minutes?) as well as to provide a powerful tool for transportation agencies to coordinate the public transportation vehicles. However, predictive analysis of the performance reliability of public transportation vehicles is challenging because a major factor impacting the (near) future performance reliability of public transportation vehicles is traffic congestion in the future. Data-driven approaches that use big datasets collected from transportation systems (e.g., bus trajectories, traffic sensors, accident logs) offer a unique opportunity to mine and understand the factors causing traffic congestions and in turn, help to forecast the performance reliability of public transportation vehicles.

In this project, we developed a data-driven, deep learning approach for traffic flow forecasting and bus arrival time estimation. Specifically, we developed a novel Graph Convolutional Recurrent Neural Network (GCRNN) to model and forecast traffic flows at various spatial (e.g., individual regions, road segments, or sensors) and the temporal (e.g., next 5 minutes and 30 minutes) resolutions and achieved the best performance for all forecasting horizons compared to the existing methods. We also implemented a Geo-Convolution Long Short-Term Memory (Geo-Conv LSTM) framework to model bus ETA by incorporating the traffic flow predictions of our GCRNN. Using the real-world traffic sensor datasets archived in our data warehouse, we showed that our proposed bus ETA model is more accurate than the existing method, Gradient Boosted Decision Tree (GBDT), by 27% in estimating bus travel times. Lastly, we deployed both models as web applications so that users can access traffic prediction data and check bus arrival times to a destination location from a starting point.

2. Traffic Prediction

Traffic forecasting has been studied for decades and knowledge and data driven approaches are the two main approaches being used. In knowledge-driven methods, the role of existing knowledge and well-established theory is very important for designing a model [Cascetta+13] while data-driven methods let a model find its own rules or patterns based on big data. For example, vehicle traffic patterns are highly regular on a weekly period but can deviate unexpectedly in certain situations such as inclement weather, accidents, road work, etc. A purely data-driven approach would simply ask the model to handle all the different situations, while knowledge-driven systems will take advantage of existing knowledge to help guide the machine learning process. Auto-Regressive Integrated Moving Average (ARIMA) [Liu+11] model was the most statistically significant data-driven method for traffic forecasting and many variants of ARIMA were proposed. However, these models usually rely on the stationarity assumption and they perform reasonably well during normal operating conditions but do not respond well to external system changes. Most recently, deep learning, which is a type of machine learning method, has drawn a lot of academic and industrial interests and has been applied with success in classification tasks, object detection, and so on. Deep learning algorithms use multi-layer architectures to extract inherent features in data from the lowest level to the highest level, and they can discover huge amounts of structure in the data. As a traffic flow process is complicated in nature, deep learning algorithms represent traffic features without prior knowledge and have shown good performance for traffic flow prediction [Lv+15] [Yu+17]. However, most of them didn't consider the spatial structure of road networks. [Wu+16] and [Ma+17] model the spatial correlation with Convolutional Neural Networks (CNN), but the spatial structure is in the Euclidean space (e.g., 2D images). [Bruna+14] and [Defferrard+16] studied graph convolution, but only for undirected graphs. In this work, we represent the pair-wise spatial correlations between traffic sensors using a directed graph whose nodes are sensors and edge weights denote proximity between the sensor pairs measured by the road network distance. Then, we study the design of a deep recurrent neural network model to address temporal dynamics and spatial dependency in the graph in traffic forecasting. Finally, we compare our method with the existing work.

FIGURE 1. GRAPH CONVOLUTIONAL RECURRENT NEURAL NETWORK FOR TRAFFIC FLOW PREDICTION



2.1 Methodology

Our proposed idea consists of two phases. In the first phase, we focus on how to combine temporal sequences (i.e., time-series) and spatial characteristics (i.e., topological dependency) of sensors to Recurrent Neural Networks (RNN). In the second phase, we study how to enable event-based and long-term traffic flow prediction.

Recurrent Neural Network is a popular architecture of Neural Network which is used extensively with use cases consist of sequential data; RNN feeds back the output of the previous time frame to the next time frame in the network. Suppose the output of the network at t=1 is h0, while training the network at t=2 we will also consider h0, the output received from the previous instance of time. This property makes it very well suited to model temporal features, such as frames in a magnitude spectrogram or feature vectors in an activity matrix, by being trained to predict the output at the next time step given the previous ones. Therefore, we leverage the inherited architecture of RNN for temporal modeling. To model localized spatial dependency, we integrate graph convolutional in the state transition in the RNN to incorporate the underlying sensor network structure. Figure 1 shows the overall architecture of our proposed convolution RNN design. Graph convolution is firstly used to extract hierarchical features from the input time series, i.e., graph signals. These features are able to capture the topological dependency by exploiting the underlying sensor locations. Then the extracted features are fed into a deep RNN encoder-decoder pipeline. Both the parameters of the graph convolution and the RNN are learned jointly from raw time series data in an end-to-end manner.

2.1.1 Spatial Dependency Modeling

The traffic time series data demonstrate strong spatial/topological dependency, i.e., sensors that are close together tend to be related in terms of speed and number of cars passing over them than sensors that are apart. This is mainly due to (1) network and intersection connectivity and (2) flow conservation, i.e., the number of vehicles entering and exiting the road segments are related. We capture these observations by working on the following subtasks.

Graph Attention Mechanism

Recurrent Neural Network feeds back the output of the previous time frame to the next time frame in the network continuously and we can say that the RNN considers all the historical observations, which is a big advantage of using RNN to capture continuous temporal dependencies. Yet, the spatial dependency of traffic is rather localized; traffic sensors that are close together tend to have strong correlations. To account for such local dependency, we adopt an attention mechanism for spatial modeling. Attention mechanism simply puts more weight to more related components. To let our model learn the attention mechanism, we train the model to focus only on the close neighborhood instead of the entire road network. Then, we take a weighted combination of the hidden states from nearby sensors weighted by the attention; nearby sensors will have higher weight values. The attention mechanism is defined as:

$$f_{att}(\mathbf{h}_i, \mathbf{h}_j) = \mathbf{h}_i^\top \mathbf{W_a} \mathbf{h}_j, \quad a_{ij} = \frac{\exp(f_{att}(\mathbf{h}_i, \mathbf{h}_j))}{\sum_{k \in nb(i,K)} \exp(f_{att}(\mathbf{h}_i, \mathbf{h}_k))}, \quad \mathbf{g}_i = \sum_{j \in nb(i,K)} a_{ij} \mathbf{h}_j,$$

Where h_i denotes the hidden states of sensor i which is extracted using an RNN shared across all the nodes. $nb(i, K)$ will return the set of neighbors that are within K-hop from node i, and g_i represents the aggregated hidden state for node i that incorporates information from neighborhood nodes. Consequently, the forecasting task of node i will be implemented using a fully connected feed-forward network with g_i as the input.

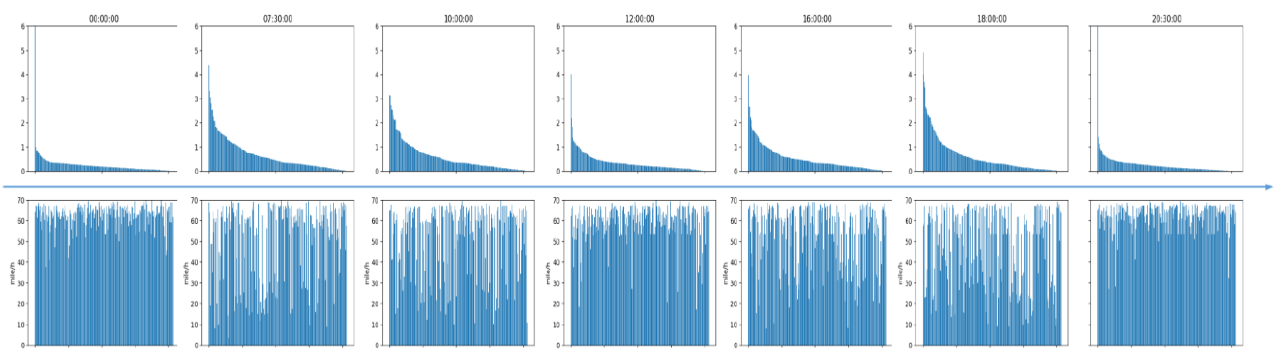
Graph Laplacian Transformation

Graph attention mechanism enables local dependency-based road network structure modeling, but in practice, it only provides marginal performance gain. This is partly because graph attention only models the topological dependency in the vertex (traffic sensor) domain, and yet it fails to capture the “conservation of flow” property in traffic. To resolve this issue, we transform the graph representation of road networks with traffic sensors into a spectral domain using Graph Laplacian; the transformation to a spectral domain of graphs is a commonly used method to better represent the characteristics of the graph. Graph Laplacian is well known to provide insight on diffusion (in our case, traffic flow diffusion) in the vertex domain. For instance, applying the Laplacian operator (L) to a signal x represents a one-step diffusion of the signal on the graph. We can model the traffic flow change as $\frac{\partial x_i(t)}{\partial t} = cL_i x$ with L_i as the i-th row vector of the graph Laplacian, and c as some constant. This transformation is known as the graph convolution kernel, denoted as *g. To obtain the Laplacian matrix, we construct the adjacency matrix based on road network distances with a threshold Gaussian kernel [SNF+03]. The k-th power of Laplacian is supported by the k-hop neighbors [SNF+03] representing the spread of traffic flow at a different scale. To model the spatial dependency at a different resolution, we compute a weighted sum of the k-th power of Laplacian as the spectral transformation. Computing the k-th power Laplacian matrix can be computationally expensive, so we apply the Chebyshev polynomial expansion for efficient approximation; One can obtain polynomials very close to the optimal one by expanding the given function in terms of Chebyshev polynomials and then cutting off the expansion at the desired degree.

2.1.2 Temporal Dynamics Modeling

We model the temporal dynamics by leveraging the inherited attributes of RNN. In particular, for temporal dynamics modeling, we use the Gated Recurrent Units (GRU) [CGC+14] – one of the variants of RNN - which has a simple structure and competitive performance. We incorporate spatial dependencies into GRU by replacing the matrix multiplication with the graph convolution $*_{\mathcal{G}}$ defined in section 2.1.1. This graph convolutional operation is applied to both inputs and hidden states to obtain a Graph Convolutional Gated Recurrent Unit (GCGRU). We stack GRU and unroll the recurrence for a fixed number of steps $\boldsymbol{T}$ and use backpropagation through time in order to compute gradients. Figure 2 shows the road network traffic evolution in 24 hours, going through morning rush hour and afternoon rush hour. We observe that in the spectral domain, the traffic speed time series enjoys better sparsity than in the vertex domain. This means that the distribution of the transformed input reflects the traffic congestion condition. With heavy congestion in rush hours, the spectral distribution of the time series becomes more heavy-tailed.

Figure 2. Visualization of 24 hours road network traffic time series evolution in spectral domain with Laplacian transformed input (top row) and vertex domain with raw input (bottom row)



2.1.3 Event based and Long-Term Forecasting

We believe that - particularly during long-term forecasting- simply training a model for one step ahead prediction and then back feeding the predictions at test time is prone to large error propagation. The forecasting error in earlier steps could be quickly amplified over a long-time span. To predict traffic flows in case of events and in long terms (e.g., 1 day in the future), we leverage encoder-decoder architecture [SVL+14] as well as scheduled sampling [BVG+15].

In particular, we first feed the historic time series data into a deep RNN encoder and generate final states. Then, we use the final states of the encoder as the initial states for a deep RNN decoder, which generates the future time series given the current state of the model and the previous ground truth target. In test time, ground truth observations become unavailable and are thus replaced by predictions generated by the model itself. The entire encoder-decoder model is trained by maximizing the likelihood of generating the target future time series given the input. One issue of this approach is the discrepancy in input distribution during training and testing. In training, the model only learns to make predictions given the ground truth observations from the last step; however, in testing the model is required to deal with its own mistakes made in previous predictions. To mitigate the issue, we use a scheduled sampling approach into the model. Scheduled sampling can be considered as a regularization method to prevent the model from overfitting and randomly replaces inputs to the decoder with model predictions during the course of training. For instance, a decoder is first trained with the previous ground truth target. After a certain amount of iterations, the decoder is fed as the input either the output of the encoder with probability p or the true previous ground-truth value with probability (1- p).

2.2 Evaluation

We conducted experiments on real-world large-scale datasets: METRA-LA. This traffic dataset contains traffic information collected from loop detectors on the highway of Los Angeles County [Jagadish+14]. We selected 207 sensors and collected 4 months of data ranging from Mar 1st, 2012 to Jun 30th, 2012 for the experiment. We aggregated traffic speed readings into 5 minutes of windows and apply Z-Score normalization. 70% of data was used for training, 20% was used for testing while the remaining 10% for validation. We compared our traffic forecasting model with widely used time series regression models, including (1) HA: Historical Average, which models the traffic flow as a seasonal process, and uses weighted average of previous seasons as the prediction; (2) ARIMA: Auto-Regressive Integrated Moving Average model with Kalman filter which is widely used in time series prediction; (3) VAR: Vector Auto-Regression (Hamilton, 1994). (4) SVR: Support Vector Regression which uses linear support vector machine for the regression task; The following deep neural network-based approaches are also included: (5) Feed-forward Neural network (FNN): Feed-forward neural network with two hidden layers and L2 regularization. (6) Recurrent Neural Network with Fully Connected LSTM hidden units (FC-LSTM) [Sutskever+14]. All neural network-based approaches were implemented using Tensorflow [Abadi+16] and trained using Adam optimizer with learning rate annealing. The best hyperparameters were chosen using the Tree-structured Parzen Estimator (TPE) [Bergstra+11] on the validation dataset.

Table 1. Performance comparison of different approaches for traffic speed forecasting

Dataset	Time	HA	ARIMA	VAR	SVR	FNN	FC-LTM	GCRNN
METRA LA	15 min	13.0%	9.6%	10.2%	9.3%	9.9%	9.6%	7.3%
	30 min	13.0%	12.7%	12.7%	12.1%	12.9%	10.9%	8.8%
	1 hour	13.0%	17.4%	15.8%	16.7%	14.0%	13.2%	10.5%

Table 1 shows the comparison of different approaches for 15 minutes, 30 minutes and 1 hour ahead forecasting. These methods were evaluated based on a commonly used metric in traffic forecasting, Mean Absolute Percentage Error (MAPE); MAPE is a statistical measure of how accurate a forecast system is. It measures this accuracy as a percentage and can be calculated as the average absolute percent error for each time period minus actual values divided by actual values. Models with lower MAPE values are better. Missing values were excluded in calculating these metrics. We observed the following phenomenon. (1) RNN-based methods, including FC-LSTM and GCRNN, generally outperform other baselines which emphasizes the importance of modeling the temporal dependency. (2) GCRNN achieved the best performance for all forecasting horizons, which suggests the effectiveness of spatiotemporal dependency modeling. (3) Deep neural network-based methods including FNN, FC-LSTM, and GCRNN, tend to have better performance than linear baselines for long-term forecasting, e.g., 1 hour ahead. This is because the temporal dependency becomes increasingly non-linear with the growth of the horizon. Besides, as the historical average method does not depend on short-term data, its performance is invariant to the small increases in the forecasting horizon.

3. Bus Arrival Time Estimation

Due to the traffic congestion problems, increasing ridership of public transportation has been one of the top primary objectives for transportation agencies and policymakers. Historical performance measurements of public transportation systems can help identify problems and potential solutions for improving ridership. Beyond historical performance measurements, accurate predictive analysis of performance reliability helps to manage rider expectations as well as to provide a powerful tool for transportation agencies to coordinate the public transportation vehicles.

We utilize the traffic flow prediction results of our GCRNN to improve the performance reliability of public transportation vehicles, especially buses in Los Angeles. First, we implemented Geo-Convolutional Long Short Term Memory (Geo-Conv LSTM) based bus arrival time estimation model. Second, we incorporated the traffic prediction results into our bus ETA model. Finally, we compare our method with the existing work in predicting bus arrival times.

3.1 Methodology

We first focus on learning the spatial and temporal dependencies from the raw GPS points of bus trajectories. Second, we study how to utilize speed values for predicting bus travel times. Third, we investigate how to integrate external factors such as day of the week and personal/driver information. Finally, we work on optimizing the methodology to predict the travel time in a long distance as well as in a short distance by utilizing multi-task learning. Our model is shown in Figure 3 and we explain each feature in more detail in the following sections.

3.1.1 Spatial Dependency Modeling

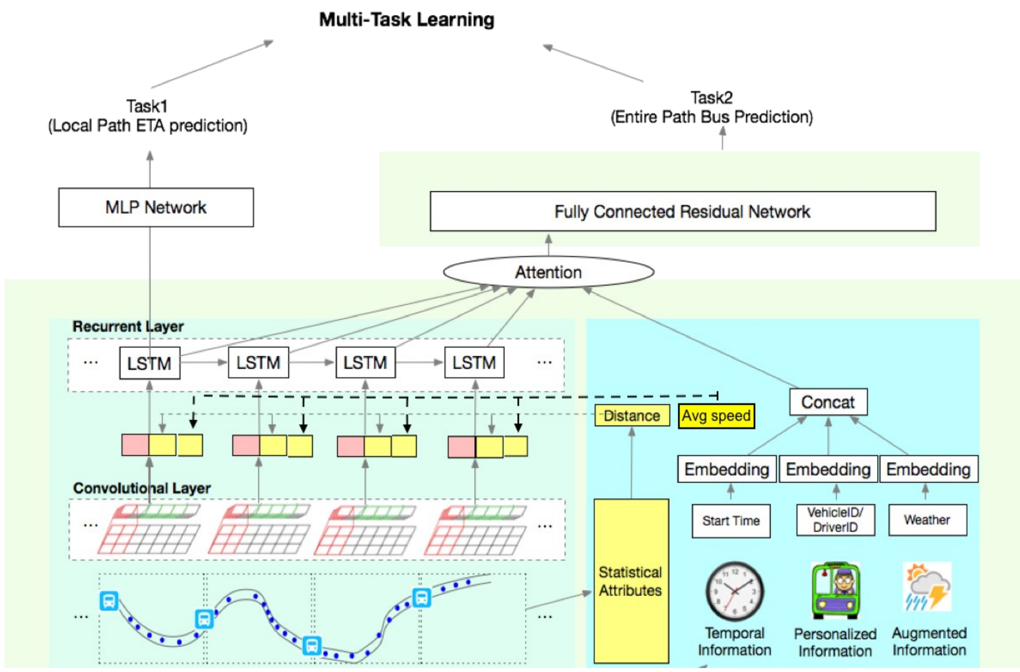
Capturing the spatial dependencies in the GPS sequence of bus trajectories is critical to travel time estimation. A standard technique to capture the spatial dependencies is the Convolution Neural Network (CNN); for instance, 2D-CNN partitions an area into *I* x *J* grids and maps each GPS coordinate into a grid cell. However, directly mapping the GPS coordinates into a grid cell is not accurate enough to represent the original spatial information in the data. For example, we cannot distinguish the turnings if the related locations are mapped into the same cell. Geo-Convolution (Geo-Conv) [DON+18] was introduced to capture the spatial dependency in the geo-location sequence while retaining the information in fine granularity.

More specifically, Geo-Conv converts a pair of longitude and latitude of a GPS point into a point in the 16-dimensional space. For each GPS point p_i in a bus trajectory, it is a non-linear mapping: $loc_i = \tanh(W_{loc} \,.\, [p_i.lat \oplus p_i.lon]$ where $\oplus$ is the concatenate operation and W_{loc} is a learnable weight matrix. The output sequence $loc \in R^{16\times n}$ represents mapped locations. We apply a convolution operation on the sequence loc along with a 1-dimensional sliding window: $loc_i^{conv} = \delta^{cnn}(W_{conv} * loc_{i:i+k-1} + b)$ where * is the convolution operation, δ^{cnn} is an activation function, $loc_{i:i+k-1}$ is a subsequence of loc, and b is a bias term. This method is known to be more useful to capture spatial dependencies of GPS points than traditional CNN (refer [DON+18] for more detail).

3.1.2 Temporal Dynamics Modeling

To capture the temporal dependencies among bus trajectories/paths, we introduce the recurrent layer in our model. The recurrent neural network (RNN) is an artificial neural network that is widely used for capturing temporal dependency. In particular, we use LSTM – one of the variants of RNN - which is known to overcome the gradient vanishing problem of RNN and captures the temporal dependencies of long sequences. In our model, we stack two LSTM layers on top of one Geo-Conv layer.

FIGURE 3. GEO-CONVOLUTION LONG SHORT TERM MEMORY NETWORK FOR BUS TRAVEL TIME PREDICTION



3.1.3 Traffic Flow Incorporation

As mentioned in the introduction, we use traffic prediction values of our GCRNN model to improve the performance of our bus ETA model. However, our GCRNN model predicts speed values of traffic sensors given as inputs and it doesn't provide speed values of GPS points in bus trajectories. To estimate the speed value of a GPS point in a bus trajectory, we calculate a weighted average of its neighboring sensors' speed values. More specifically, for each location point, we select k nearest traffic sensor and weight-average the predicted speed values of the sensor locations. The weights are computed by inversing distance values from the GPS location to the neighboring traffic sensor locations; closer sensors have higher weights than farther sensors. Figure 4 illustrates the speed estimation process when k is 2. Let o be a GPS point we want to estimate speed from its k neighboring traffic sensors $s_1, s_2, \dots, s_k$ with their corresponding distances to the GPS point: $dn_1, dn_2, \dots, dn_k$. In this work, the distance between GPS points is a geographical distance, the distance measured along the surface of the earth, between two pairs of latitude-longitude coordinates. The speed at o is calculated as follows:

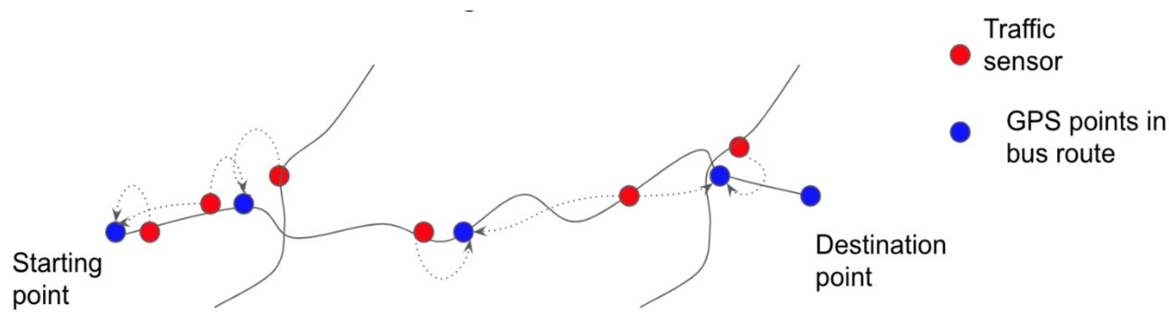
$$speed(o) = \frac{\frac{s_1}{dn_1} + \frac{s_2}{dn_2} + \cdots + \frac{s_k}{dn_k}}{\frac{1}{dn_1} + \frac{1}{dn_2} + \cdots + \frac{1}{dn_k}}$$

In addition, we use local distance information, a distance from the starting point to a GPS point in a trajectory. For each GPS point p_i, we concatenate its geo-convolution output with a local distance, a current speed and predicted speed values in the next t time steps. An input vector to LSTM, lsd_i, now consists of a geo-convolution output of a GPS coordinate, a current speed, speed values in next t time steps and a local distance:

$$lsd_i = loc_i^{conv} \oplus s_i \oplus d_i$$

where loc_i, s_i, d_i are vectorized location values, speed sequences (current speed and predicted speed in next t steps), a local distance value of p_i respectively, and $\oplus$ is a concatenation operation.

Figure 4. Example of estimating speed at GPS points using two nearby traffic sensors



3.1.4 Context Aware Travel Time Modeling

The travel time of a path is affected by many complex factors, such as the start time, the day of the week, the weather condition and the driving habits. Bus trajectory data in our repository includes timestamp, vehicle ID and driver ID in addition to location information. These data are categorical values, which cannot feed directly to the neural network. Each categorical attribute is represented by a one-hot vector; the size of the vector is the number of possible categories for each attribute and only the category a value belongs to set to 1 (the rest vector values are 0). All one-hot vectors are concatenated.

3.1.5 Model Optimization

There mainly exist two approaches to estimate the travel time of a path: (1) Individual travel time estimation that firstly splits a path into several road segments and estimate the travel time for each local path, finally sums over them to get the total travel time and (2) Collective travel time estimation that directly estimates the travel time of the entire path. If we adopt the individual estimation, the local errors may accumulate since such method does not consider the spatio-temporal dependencies among the local paths. In the meantime, if we use the collective estimation, we usually face the data sparsity problem since only a few trajectories traveled through the entire path or the longer sub-paths. To prevent this, we use multi-task learning to combine these two methods. Let L_1 be the loss of the individual estimation and L_2 be the loss of the collective estimation. The loss that the model minimizes is

$$L = \alpha \times L_1 + (1 - \alpha) \times L_2$$

The parameter α is the combination coefficient that linearly balances the tradeoff between L_1 and L_2. In our experiment, the parameter α was set to 0.05. During the training phase, we enforce the multi-task learning component to accurately estimate the travel time of both the entire path and each local path simultaneously. During the test phase, we eliminate the local path estimate part and report the estimated travel time of the entire path.

3.2 Evaluation and Prototype Development

At USC's IMSC with our partnership with Los Angeles Metropolitan Transportation Authority (LA Metro) and METRANS, we developed a big transportation data warehouse-Archived Traffic Data Management System (ADMS). ADMS fuses and analyzes a very large-scale and high-resolution (both spatial and temporal) traffic sensor data from different transportation authorities in Southern California, including California Department of Transportation (Caltrans), Los Angeles Department of Transportation (LADOT), California Highway Patrol (CHP), Long Beach Transit (LBT). This data set includes both inventory and real-time data with update rate as high as every 30 seconds for freeway and arterial traffic sensors (14,500 loop-detectors) covering 4,300 miles, 2,000 bus, and train automatic vehicle location (AVL), incidents such as accidents, traffic hazards and road closures reported (approximately 400 per day) by LAPD and CHP, and ramp meters. We have been continuously collecting and archiving datasets for the past 5 years. ADMS, with 11TB annual growth, is the largest traffic sensor data warehouse built so far in Southern California. We used the aforementioned real-world data to train and evaluate our approach for estimating the travel time of an individual bus given a trajectory/path and a start time. We compared our results with the existing work. We also deployed the new capabilities developed in this project as a web service. RTX 2090 Ti GPU (boost clock 1545 MHz, memory speed: 14Gbps) was used to train and evaluate the models.

3.2.1 Experiments

We selected 62,000 trajectories from our data repository collected for two months (April 2017, September 2017) in Los Angeles. This data contains location (longitude and latitude), timestamp, and vehicle ID information, which allows us to compute ground truth speed values of each GPS point and travel times of bus stops for evaluation. 50000 trajectories out of 62000 were used for training the Geo-Conv LSTM bus ETA model and the rest was used for evaluation. The mean of each trajectory travel time is 1,344sec. The distance mean is 21km; most of the trajectories in the test data are shorter than 20 km. Figure 5 shows the GPS points of all trajectories.

We compared our bus ETA model with commonly used travel time estimation methods including (1) AVG: The estimated travel time = Total distance of the path / Average speed of entire network at the Starting hour in the past - training data, (2) Linear Regression (LR) [PED+11]: The estimated travel time = Linear function f of input features, (3) Support Vector Regression (SVR) [PED+11]: f is a support vector regression model and (4) Gradient Boosting Decision Tree (GDBT) [PED+11]: f is learned based on the boosting of decision tree models. Table 2 shows

Figure 5. Bus trajectories in LA dataset

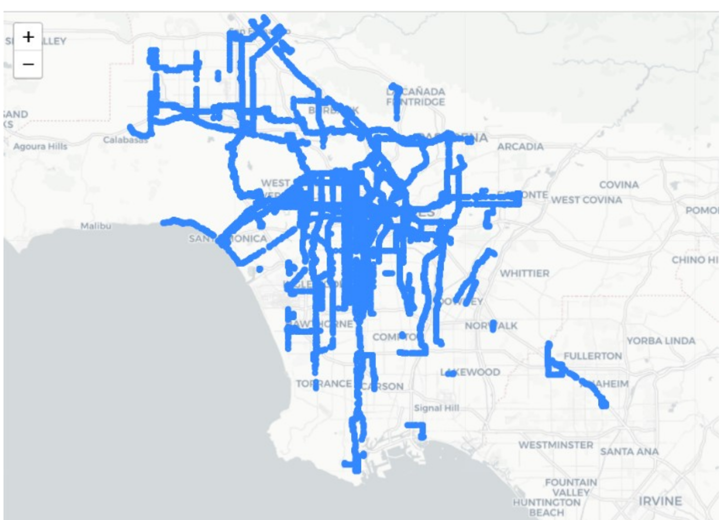


Table 2. MAPE of Bus ETA methods using LA dataset

Model	MAPE (%)
AVG	51
LR	79
SVR	39
GBDT	36
Geo Conv LSTM without traffic prediction	33
Geo Conv LSTM with traffic prediction	26

Table 3. MAPE with varying the number of neighboring sensors for speed estimation

#Neighboring sensors	**MAPE (%)**
No traffic prediction data (baseline)	6.3
1 neighbor	8.0
3 neighbors	7.0
5 neighbors	5.2
10 neighbors	5.9

Table 4. MAPE Comparison of speed estimation with different weighted averaging methods

Estimation Method	**MAPE (%)**
Equal Weight Average	6.2
Distance-based Weighted Average	5.2

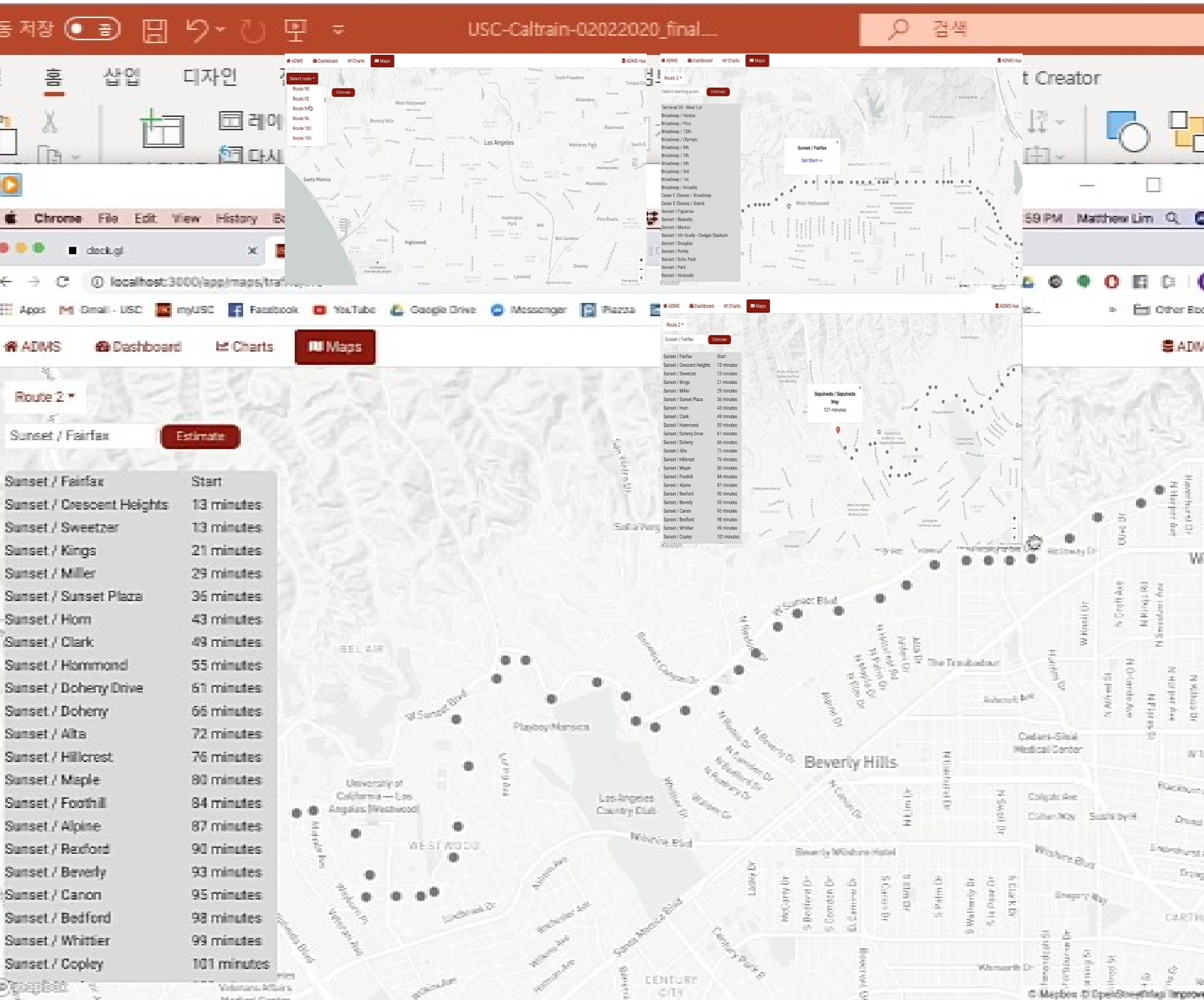
the comparison of different approaches for estimating travel times. These methods were evaluated based on a commonly used metric in traffic forecasting, Mean Absolute Percentage Error (MAPE). We observed the following phenomena. (1) RNN-based methods (Geo-Conv LSTM model) generally outperform other baselines which emphasizes the importance of modeling the temporal dependency. (2) Geo-Conv LSTM with traffic prediction achieves the best performance by 6% MAPE comparing to Geo-Conv LSTM without traffic prediction, which suggests the effectiveness of utilizing the traffic forecasting results.

As explained in section 3.1.3, to estimate the speed value of a GPS point in a bus trajectory, we calculate a weighted average of its neighboring sensors' speed values. To show the effectiveness of our approach, we compared the performance of estimating travel times in two different conditions: (1) varying the number of neighboring sensors from 1 to 10 (2) using an equal weight vs an inverse distance-based weight for averaging speed values. We only used a small dataset for these experiments, 1-day bus trajectories in Los Angeles.

Table 3 shows that when 5 neighboring sensors were used, our bus ETA approach worked the best with 17.4% improvement. We observed that MAPE is even worse when the number of neighboring sensors is 1 and 3 than the model without speed information. It is difficult to estimate the speed of a GPS point with a small number of neighboring sensors. When we increased the number of neighboring sensors to 10, the MAPE increased because more unrelated sensors were included for speed estimation. Table 4 shows that the inverse distance-based weighted average approach produces better performance. We plan to run the same experiments with a larger dataset.

3.2.2 Web Application

We deployed our bus ETA model as a web service and it's currently hosted at an Amazon server (http://40.117.179.70:3000/app).We're working on adding this service to our existing web application (adms.usc.edu) and the web service will be publicly available. The purpose of our web service is to let users plan a trip before ahead with the information of predicted bus arrival times to any bus stops on a selected bus route. Figure 6 shows the prototype of the dashboard. Using this web service, for example, a user can select a bus route from a dropdown menu (Fig 6. top left). The bus route will be shown on the map as well as the names of all the bus stops on the selected route (Fig 6. top right) and the user can select a start bus stop (optionally a start time). Then Estimated Time of Arrival (ETA) to all the bus stops on the route will be shown on the left sidebar (Fig 6. bottom left) and of course, the ETA of bus stops will be shown in a massage box if the user clicks a bus stop on the map. the user can see the ETA to the destination bus stop (Fig 6. bottom right). Then the user can figure out when would be the best time for him/her to start his/her trip and can also share the arrival time information with friends/family/co-workers. To make this web service up and run, we designed and implemented the back-end systems as shown in figure 7, the web prototype of our bus ETA system consists of five main components:

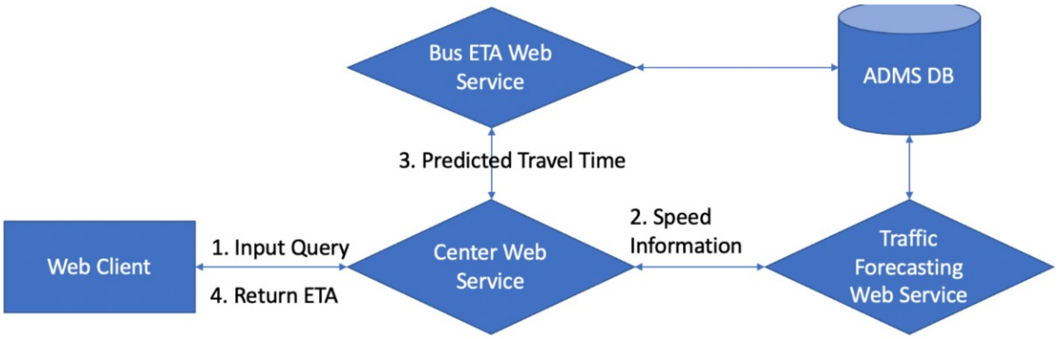


- Bus ETA Web Service: This contains our Geo-Conv LSTM based Bus ETA model. It receives an input query from Web Client and returns the predicted travel time of the destination.

- Traffic Forecasting Web Service: This has our GCRNN traffic forecasting model. It predicts the speed of a traffic sensor in k time steps.

- ADMS Database: This is our traffic data warehouse storing traffic data, traffic sensor data, bus trajectory, bus route, and vehicle information data, etc.

FIGURE 7. BUS ETA WEB APPLICATION SYSTEM DESIGN



4. Conclusion

Our data warehouse - Archived Traffic Data Management System (ADMS), with 11TB annual growth, is the largest traffic sensor data warehouse built so far in Southern California. Using this big traffic dataset, we have a unique opportunity to use data-driven approaches to understand the factors causing traffic congestions and in turn, help to forecast the performance reliability of public transportation vehicles. In this paper, we proposed a reliability analysis system using Deep Learning (DL) techniques to forecast the future performances of the public bus system in Los Angeles. More specifically, we designed and developed a GCRNN traffic forecasting model that captures spatial and temporal dependencies as well as long-term forecasting. In turn, we incorporated the traffic prediction results of our GCRNN model to the Geo-Conv LSTM bus ETA model and outperform the baseline model (GBDT) by 27% in estimating travel times. The system demonstrates the overall approach in an area near downtown Los Angeles and shows that incorporating traffic flow predictions can help to forecast short-term bus arrival times accurately (e.g., in the next few hours).

Although our experimental results show significant improvement compared to the state-of-the-art baselines methods, modeling Deep Learning (DL) techniques to forecast the future performances of the public transportations for large spatial scale (e.g., the entire Los Angeles Metropolitan Area) and long-term (e.g., days instead of hours) remains challenging. Reliable long-term forecasting of performance measurement for public transportation systems over a large area is essential for policymakers to achieve effective city planning as well as promotes ridership. For example, forecasting bus arrival time for the next day helps a rider to plan their commute early. Existing approaches typically rely on traffic simulation tools and models that require expert knowledge to execute and adjust parameters for various traffic scenarios. We want to expand our current approach and system to develop the capability for processing the entire Los Angeles Metropolitan Area for long-term forecasting of a variety of public transportation system performance metrics.

References

[Abadi+16] Tensorflow: Large-scale machine learning on heterogeneous distributed systems. arXiv preprint arXiv:1603.04467, 2016.

[Bergstra+11] James S Bergstra, Remi Bardenet, Yoshua Bengio, and Balazs Kegl. Algorithms for hyper-parameter, In NIPS, 2011.

[Bruna+14] Joan Bruna, Wojciech Zaremba, Arthur Szlam, and Yann LeCun. Spectral networks and locally connected networks on graphs. In ICLR, 2014.

[BVJ+15] S. Bengio, O. Vinyals, N. Jaitly, and N. Shazeer. Scheduled sampling for sequence prediction with recurrent neural networks. In NIPS, 2015.

[Cascetta+13] Transportation systems engineering: theory and methods, volume 49. Springer Science & Business Media, 2013.

[CGC+14] J. Chung, C. Gulcehre, K. Cho, and Y. Bengio. Empirical evaluation of gated recurrent neural networks on sequence modeling. arXiv preprint, 2014.

[Defferrard+16] Michael Defferrard, Xavier Bresson, and Pierre Vandergheynst. Convolutional neural networks on ¨ graphs with fast localized spectral filtering. In NIPS, pp. 3837–3845, 2016.

[DON+18] Wang, Dong, et al. "When will you arrive? estimating travel time based on deep neural networks." *Thirty-Second AAAI Conference on Artificial Intelligence*. 2018

[DSU+16] D. Deng, C. Shahabi, U. Demiryurek, L. Zhu, R. Yu, and Y. Liu, Latent Space Model for Road Networks to Predict Time-Varying Traffic, In SIGKDD, 2016.

[EST+80] Dagum, Estela Bee. *The X-II-ARIMA seasonal adjustment method*. Statistics Canada, Seasonal Adjustment and Time Series Staff, 1980.

[HSH+15] W. Huang, G. Song, H. Hong and K. Xie, Deep architecture for traffic flow prediction: Deep belief networks with multitask learning. In IEEE Transactions on Intelligent Transportation Systems, 2014.

[Jagadish+14] H. V. Jagadish, Johannes Gehrke, Alexandros Labrinidis, Yannis Papakonstantinou, Jignesh M. Patel, Raghu Ramakrishnan, and Cyrus Shahabi. Big data and its technical challenges. Commun. ACM, 57(7):86–94, July 2014

[Liu+11] Wei Liu, Yu Zheng, Sanjay Chawla, Jing Yuan, and Xie Xing. Discovering spatio-temporal causal interactions in traffic data streams. In SIGKDD, pp. 1010–1018. ACM, 2011.

[Lippi+13] Marco Lippi, Marco Bertini, and Paolo Frasconi. Short-term traffic flow forecasting: An experimental comparison of time-series analysis and supervised learning. ITS, IEEE Transactions on, 14(2): 871–882, 2013.

[LDK+15] Y. Lv, Y. Duan, W. Kang, Z. Li, and F.-Y. Wang, Traffic flow prediction with big data: A deep learning approach. In IEEE Transactions on Intelligent Transportation Systems, 2015.

[Lv+15] Yisheng Lv, Yanjie Duan, Wenwen Kang, Zhengxi Li, and Fei-Yue Wang. Traffic flow prediction with big data: A deep learning approach. ITS, IEEE Transactions on, 16(2):865–873, 2015.

[Ma+17] Xiaolei Ma, Zhuang Dai, Zhengbing He, Jihui Ma, Yong Wang, and Yunpeng Wang. Learning traffic as images: a deep convolutional neural network for large-scale transportation network speed prediction. Sensors, 17(4):818, 2017.

[PED+11] Scikit-learn: Machine Learning in Python, Pedregosa *et al.*, JMLR 12, pp. 2825-2830, 2011.

[Sutskever+14] H. V. Jagadish, Johannes Gehrke, Alexandros Labrinidis, Yannis Papakonstantinou, Jignesh M. Patel, Raghu Ramakrishnan, and Cyrus Shahabi. Big data and its technical challenges. Commun. ACM, 57(7):86–94, July 2014

[YAG+17] Li, Yaguang, et al. "Diffusion convolutional recurrent neural network: Data-driven traffic forecasting." *arXiv preprint arXiv:1707.01926* (2017).

[YLS+17] R. Yu, Y. Li, C. Shahabi, U. Demiryurek, Y. Liu, Deep Learning: A Generic Approach for Extreme Condition Traffic Forecasting. Proceedings of the 2017 SIAM International Conference on Data Mining (SDM), 2017

[Yu+17] Rose Yu, Yaguang Li, Cyrus Shahabi, Ugur Demiryurek, and Yan Liu. Deep learning: A generic approach for extreme condition traffic forecasting. In SIAM International Conference on Data Mining (SDM), 2017b

[Wu+16] Yuankai Wu and Huachun Tan. Short-term traffic flow forecasting with spatial-temporal correlation in a hybrid deep learning framework. arXiv preprint arXiv:1612.01022, 2016.

Data Management Plan

Data Description

Our partnerships with Los Angeles Metropolitan Transportation Authority (LA Metro) enables our data repository, ADMS, to access large and high-resolution (both spatial and temporal) traffic sensor data from a number of transportation authorities in Southern California. This dataset includes both sensor metadata and real-time data for freeway and arterial traffic sensors (~16,000 loop-detectors) covering 4,300 miles, 2,000 bus and train automatic vehicle locations (AVL), and ramp meters with an update rate as high as 30 seconds. The dataset also includes data about several incident types (such as accidents, traffic hazards, and road closures) reported by LAPD and CHP at a rate of approximately 400 incidents per day. We have been continuously collecting and archiving the aforementioned datasets for the past 8 years and with an annual growth of 1.5TB, ADMS is the largest traffic sensor data warehouse in Southern California.

Data Format and Content

We detail the schema and the attribute of each database table in our repository.

Table Name: CONGESTION_INVENTORY

- Congestion: Data and metadata related to the road network and the congestion.
- Metadata: Static information about the sensors, i.e., loop detectors.

Column	Description
AGENCY (string)	Agency that provided the record
CITY (string; nullable)	City name
DATE_AND_TIME (timestamp)	Date and time
LINK_ID (string)	Sensor ID
LINK_TYPE (enum)	Sensor type (HIGHWAY, ARTERIAL)
ON_STREET (string)	Street that sensor is on
FROM_STREET (string; nullable)	
TO_STREET (string; nullable)	
START_LOCATION (geometry)	Latitude and longitude of sensor

DIRECTION (enum)	NORTH, SOUTH, EAST, WEST, UNSPECIFIED
POSTMILE (float)	Distance from a specific end of the road
NUM_LANES (integer)	Affected number of lanes

Table Name: CONGESTION_DATA

- Data: Time-series of each sensor. It contains the occupancy, volume, and speed for each sensor at every timestep.

Column	Description
AGENCY (string)	Agency that provided the record
DATE_AND_TIME (timestamp)	Date and time
LINK_ID (sting)	Sensor ID
OCCUPANCY (float)	The percentage of time a sensor detects a vehicle in 30 seconds ● For example, an occupancy of 5% means that of those 30 seconds, vehicle presence was detected for an aggregate 1.5 seconds
SPEED (float)	Distance traveled per unit time, and in traffic operations ● mean speeds within a given roadway section (link)
VOLUME (integer)	Represents the number of vehicles that passed by per sensor every 30 seconds
HOVSPEED (float)	Speed on HOV lane(s)
LIN_STATUS (enum)	Status of sensor, “OK”, “FAILED” or “UNKNOWN”.

Table Name: RAMP_METER_INVENTORY

- Ramp Meters: Complementary to the congestion data. Ramp meters are the entry points to highways from arterial roads.

- Metadata: Static information about ramp meters.

Column	Description
RAMP_ID (integer)	Unique ramp id
AGENCY (string)	Agency that provided the record
DATE_AND_TIME (timestamp)	Date and time
CITY (string; nullable)	City name
MS_ID	
RAMP_TYPE (integer)	
ON_STREET (string)	Street that ramp is on
FROM_STREET (string; nullable)	
TO_STREET (string; nullable)	
LOCATION (geometry)	Latitude and longitude of sensor
DIRECTION (enum)	NORTH, SOUTH, EAST, WEST, UNSPECIFIED
POSTMILE (float)	Distance from a specific end of the road

Table Name: RAMP_METER_DATA

- Data: Time-series and state of each ramp meter at each timestep.

Column	Description
AGENCY (string)	Agency that provided the record
DATE_AND_TIME (timestamp)	Date and time
RAMP_ID (sting)	Sensor ID

MS_ID	
DEVICE_STATUS (integer)	
METER_STATUS (integer)	
METER_CONTROL_TYPE (integer)	
METER_RATE (integer)	
OCCUPANCY (float)	The percentage of time a sensor detects a vehicle in 30 seconds ● For example, an occupancy of 5% means that of those 30 seconds, vehicle presence was detected for an aggregate 1.5 seconds
SPEED (float)	Distance traveled per unit time, and in traffic operations ● mean speeds within a given roadway section (link)
VOLUME (integer)	Represents the number of vehicles that passed by per sensor every 30 seconds
LINK_IDS (string[])	
LINK_TYPES (string[])	
LINK_OCCUPANCIES (float[])	The percentage of time a sensor detects a vehicle in 30 seconds ● For example, an occupancy of 5% means that of those 30 seconds, vehicle presence was detected for an aggregate 1.5 seconds
LINK_SPEEDS (float[])	Distance traveled per unit time, and in traffic operations ● mean speeds within a given roadway section (link)
LINK_VOLUMES (integer[])	Represents the number of vehicles that passed by per sensor every 30 seconds
LINK_STATUSES (enum[])	Status of sensors, “OK”, “FAILED” or “UNKNOWN”.

Table Name: TRAVEL_TIMES_INVENTORY

- Travel Times: Computed travel times between sensors.
- Metadata: Defines the pair of sensors for which travel times are being computed.

Column	Description
LINK_ID (integer)	Unique link id
AGENCY (string)	Agency that provided the record
DATE_AND_TIME (timestamp)	Date and time
ROUTE_ID (integer)	Unique route id
DIRECTION (enum)	NORTH, SOUTH, EAST, WEST, UNSPECIFIED
LINK_TYPE (string)	Commonly Freeway
BEGIN_ID (integer)	Id of begin link
BEGIN_STREET_NAME (string)	
BEGIN_LOCATION (geometry)	
END_ID (integer)	Id of end link
END_STREET_NAME (string)	
END_LOCATION (geometry)	
LENGTH (double)	Length of link in kilometers.

Table Name: TRAVEL_TIMES_DATA

- Data: Travel time in minutes and average speed for each link pair.

Column	Description
LINK_ID (integer)	Unique link id
AGENCY (string)	Agency that provided the record
DATE_AND_TIME (timestamp)	Date and time
LINK_SPEED (double)	Average link speed in MPH
LINK_TRAVEL_TIME (double)	Estimated time to travel on the link in minutes.

Table Name: BUS_INVENTORY

- Bus: Static and real-time information about buses.
- Metadata: Description of bus routes.

Column	Description
ROUTE_ID (integer)	Unique bus route id.
AGENCY (string)	Agency that provided the record
DATE_AND_TIME (timestamp)	Date and time
ROUTE_DESCRIPTION (string)	Textual description of route.
ZONE_NUMBERS (int [])	Ids of zones that this route operates in.

Table Name: BUS_DATA

- Data: Tracking information for buses operating on routes.

Column	Description
BUS_ID (integer)	
AGENCY (string)	Agency that provided the record

DATE_AND_TIME (timestamp)	Date and time
ROUTE_ID (integer)	Unique bus route id.
LINE_ID (integer)	
RUN_ID (integer)	
ROUTE_DESCRIPTION (string)	Textual description of route.
BUS_DIRECTION	NORTH, SOUTH, EAST, WEST, UNSPECIFIED
BUS_LOCATION	
BUS_LOCATION_TIME	
SCHEDULE_DEVIATION	
NEXT_STOP_LOCATION	
NEXT_STOP_TIME	
NEXT_STOP_SCHEDULED_TIME	
BRT_FLAG	

Table Name: CMS_INVENTORY

- CMS: Static and dynamic information about changeable-message-signs deployed on highways.
- Metadata: Static information of the signs.

Column	Description
DMS_ID (integer)	Unique device id
AGENCY (string)	Agency that provided the record
DATE_AND_TIME (timestamp)	Date and time

CITY (string; nullable)	
ON_STREET (string)	Street that ramp is on
FROM_STREET (string; nullable)	
TO_STREET (string; nullable)	
LOCATION (geometry)	Latitude and longitude of sensor
DIRECTION (enum)	NORTH, SOUTH, EAST, WEST, UNSPECIFIED
POSTMILE (float)	Distance from a specific end of the road

Table Name: CMS_DATA

- Data: Dynamic information about the state and content of each sign at each timestep.

Column	Description
DMS_ID (integer)	Unique device id
AGENCY (string)	Agency that provided the record
DATE_AND_TIME (timestamp)	Date and time
DEVICE_STATUS (string)	OK, FAILED
STATE (string)	DISPLAY, BLANK, NO_RESPONSE, UNKNOWN
DEVICE_TIME (timestamp)	
PHASE1LINE1 (string; nullable)	
PHASE1LINE2 (string; nullable)	
PHASE1LINE3 (string; nullable)	

PHASE2LINE1 (string; nullable)	
PHASE2LINE2 (string; nullable)	
PHASE2LINE3 (string; nullable)	

Table Name: EVENT_DATA

- Events: Special events, traffic incidents, and other unplanned events. Contains information like when the event happened or is planned to happen, what agencies were involved in the resolution, etc.

Column	Description
EVENT_ID (integer)	Unique event id
AGENCY (string)	Agency that provided the record
DATE_AND_TIME (timestamp)	Date and time
ADMIN_CITY (string; nullable)	
ON_STREET (string)	Street that ramp is on
FROM_STREET (string; nullable)	
TO_STREET (string; nullable)	
LOCATION (geometry)	Latitude and longitude of sensor
DIRECTION (enum)	NORTH, SOUTH, EAST, WEST, UNSPECIFIED
ADMIN_POSTMILE (float)	Distance from a specific end of the road
TYPE_EVENT (string)	CLOSURE, INCIDENT, etc.
SEVERITY (string)	NONE, MINOR, MAJOR, NATURAL_DISASTER

EVENT_STATUS (string)	CONFIRMED, UNCONFIRMED, SCHEDULED, TERMINATED, OTHER
DESCRIPTION (string)	

Data Access and Sharing

Although the dataset is collected and maintained by IMSC at our servers, all data in our traffic data repository are owned by LA-Metro. We cannot release the data without approval from LA-Metro. We plan to arrange approval from LA-Metro for use of the data for research purposes at USC.

Appendix A : Acronym

ARIMA: Auto-Regressive Integrated Moving Average

CNN: Convolution Neural Network

DL: Deep Learning

ETA: Estimated Time of Arrival

FC-LSTM: Fully Connected LSTM

FNN: Feed-forward Neural network

GBDT: Gradient Boosted Decision Tree

GCGRU: Graph Convolutional Gated Recurrent Unit

GCRNN: Graph Convolutional Recurrent Neural Network

Geo-Conv: Geo-Convolution

Geo-Conv LSTM: Geo-Convolution Long Short-Term Memory

GRU: Gated Recurrent Units

HA: Historical Average

LR: Linear Regression

MAPE: Mean Absolute Percentage Error

RNN: Recurrent Neural Network

SVR: Support Vector Regression

TPE: Tree-structured Parzen Estimator

VAR: Vector Auto-Regression

Appendix B : Code Deliverables Links

Web Service Dashboard

https://drive.google.com/open?id=1QRdyVq_CW-85-GZjkWR5eUu9QZW77ml3

Deep Bus ETA Model

https://drive.google.com/open?id=1erSSXUuSGnbeHikNnL0lr8ykKG-1j1Mw

Instruction Documentation

https://drive.google.com/file/d/1eEmT7TmT1ZoDlp18RjeE5zdINzSmCnOb/view?usp=sharing